Junta de Andalucía

Plataforma de Tramitación w@ndA 2.5.7

Manual de Instalación y Configuración

Versión: v01r01
Fecha: 17/10/2024

HOJA DE CONTROL

Proyecto	Plataforma de Tramitación w@ndA 2.5.7		
Documento	Manual de Instalación y Configuración		
Nombre del Fichero	PTW257E_MIC_Manual_Instalación_PTw@ndA_v01r01.odt		
Autor	UTE		
Versión/Revisión	v01r01	**Fecha Versión**	17/10/2024
Aprobado por	ADA	**Fecha Aprobación**	10/10/24

REGISTRO DE CAMBIOS

Versión	Causa del Cambio	Responsable del Cambio	Área	Fecha del Cambio
v01r00	Creación del documento	UTE	UTE	10/10/24
v01r01	Corrección formato	UTE	UTE	17/10/2024

CONTROL DE DISTRIBUCIÓN

Nombre y Apellidos	Cargo	Área
Manuel Escobar Montes	Jefe de Servicio	ADA
José Ignacio Cortés Santos	Director de Proyecto	ADA
Felipe José Gallego Rivera	Director de Proyecto	ADA
María Luisa Rubio Campanario	Director de Proyecto	ADA
Alejandro Martínez Marcos	Director de Proyecto	ADA
Ignacio Gordillo Díaz	Jefe de Proyecto	UTE

ÍNDICE

1 REQUISITOS PREVIOS DE INSTALACIÓN

Para poder instalar y asegurar un buen funcionamiento de la Plataforma de Tramitación w@ndA se deben cumplir los siguientes **requisitos**:

- Máquina virtual Java versión 8. URL de descarga: http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html
- Despliegue con el Servidor de Aplicaciones WildFly 15.0.1Final URL de descarga (enlace http://download.jboss.org/wildfly/1_5_.0.1.Final/wildfly-_1_5_.0.1.Final.zip)
- Acceso a base de datos con esquema compatible Trew@ 2.6.7.1
- Base de datos Oracle soportada por Hibernate 5.4.2 (10g, 11g) para el modelo de datos propio de la Plataforma de Tramitación, el cual se dividirá en dos esquemas: uno para las tablas paramétricas y otro para las configurables..
- El servidor donde se despliegue la Plataforma de Tramitación debe tener habilitada conexión a Internet –directa o mediante proxy-, ya que Struts requiere que el servidor pueda descargarse los documentos DTD de definición sintáctica de sus ficheros XML de configuración.

Requisitos opcionales en función de los componentes w@ndA que quieran integrarse en la Plataforma de Tramitación:

- Acceso a Port@firmas, versión 3.1.0 o superior.
- Acceso a @firma 6.1, para autenticación de usuarios
- Acceso a Notific@
- Acceso a un gestor documental con los requisitos definidos por la versión de Trew@ seleccionada para el despliegue de PTw@ndA.

Adicionalmente a los componentes w@ndA puede opcionalmente requerirse el acceso a Formul@ para integrarse con el motor de formularios para mostrar tareas de manipulación de datos en caso de un desarrollo vertical que haga uso de esta posibilidad. La versión 2.5.7 de PTw@ndA es compatible con la versión 4.1.0 del motor de formularios.

En cuanto a los requisitos referentes a los clientes de la Plataforma de Tramitación, hay que cumplir los definidos por:

- La propia plataforma **@firma** en su versión v6.1 (para más información, se puede consultar el Manual de Usuario de @firma en la web Plutón de la Consejería de Justicia y Administración

Pública de la Junta de Andalucía, https://ws024.juntadeandalucia.es/pluton/adminelec/ArTec/afirma.jsp).

- El generador de escritos **WebOffice 4.x** o superior (para más información, consúltese la documentación de este componente que se distribuye en los entregables oficiales de Trew@).

En esta versión de Ptw@ndA no es obligatorio el uso de WebOffice, pudiéndose utilizar en su lugar el editor de textos javascript incluido dentro del core de la aplicación. Para ello ver parámetro de configuración ***EDITOR_TEXTOS_DEFECTO*** en el punto 3.3.3.5..... Configuración de la Aplicación.

1.1 Tabla de compatibilidades

A continuación se detalla la relación de productos con los que la Plataforma de Tramitación w@ndA ha sido probada y, por tanto, se garantiza su correcto funcionamiento.

Producto	Tipo	Versiones
Sun J2SE Runtime Environment	Máquina virtual Java	JDK 8.0 u151
WildFly	Servidor de Aplicaciones	15.0.1 Final
Oracle	SGDBR	10G, 11G

1.2 Notas de versión

Puede consultar las notas de la versión correspondiente en el documento “PTW255E_NV_Notas_Version_v01r00”, suministrado en el entregable de la aplicación.

2 PROCEDIMIENTO DE PRIMERA INSTALACIÓN

A continuación se procede a describir la instalación de la Plataforma de Tramitación en un entorno donde no haya sido instalada previamente.

Indicar que el procedimiento explicado a continuación es para la versión 2.5.7 de la Plataforma de Tramitación. No existe una migración desde las versiones 1.X.

En el caso de haber estado utilizando una versión 1.X, se requiere una nueva instalación y la adaptación de los módulos verticales desarrollados. En el manual del desarrollador se encuentra toda la información necesaria para realizar estos cambios.

Para una migración desde versiones 2.0.1rX dirigirse al apartado 4 del documento.

2.1 Instalación de componentes w@ndA complementarios

2.1.1 Instalación de Trew@ 2.6.7.1

Para proceder a la instalación de Trew@ 2.6.7.1 se remite al documento de instalación del mismo distribuido con dicho componente de w@ndA. Se hace hincapié en que este es el único componente obligatorio para poner en marcha la Plataforma de Tramitación.

2.1.2 Instalación de @Firma 6.1

Para proceder a la instalación de @Firma 6.1 se remite al documento de instalación distribuido con dicho componente de W@ndA. Hay que recordar que este componente es de uso opcional en la Plataforma de Tramitación.

En el ámbito de la Junta de Andalucía, deberá emplearse la implantación corporativa desplegada en la Consejería de Hacienda y Administración Pública.

2.1.3 Instalación de Port@firmas

Para proceder a la instalación de Port@firmas se remite al documento de instalación distribuido con dicho componente de w@ndA. Hay que recordar que este componente es de uso opcional en la Plataforma de Tramitación.

La integración de la Plataforma con este componente se configura a través de la herramienta de Administración de Trew@, en la sección de administración de componentes. Para más información se puede consultar el manual de usuario de dicha herramienta.

La plataforma de Tramitación requiere como versión mínima del componente Port@firmas la v3.1.0.

2.2 Instalación de otros componentes

2.2.1 Instalación y configuración de LibreOffice

El servidor de aplicaciones donde se encuentre el software de PTw@ndA necesita tener instalado el producto LibreOffice, para el tratamiento de algunos archivos a utilizar en la plataforma (típicamente, la conversión documentos a formato PDF). El presente apartado es una pequeña guía de instalación que resume los pasos necesarios para su instalación.

Para la instalación de LibreOffice no se necesita ningún requerimiento de hardware adicional al necesario para la ejecución del software de la Plataforma de Tramitación. Normalmente, será necesaria la intervención de un técnico de sistemas para la instalación de LibreOffice y los paquetes necesarios.

Este manual presupone que ya se tienen instalados los siguientes componentes:

- Listado de recursos software S.O Linux
  - LibreOffice (https://es.libreoffice.org/ para descarga y manual de Instalación).
- Listado de recursos software S.O Windows
  - LibreOffice (https://es.libreoffice.org/ para descarga y manual de Instalación).

Será necesario indicar al servidor de aplicaciones la ruta de instalación del LibreOffice, incluyendo un nuevo parámetro de arranque al WildFly indicando la ruta a la carpeta de LibreOffice:

office.home= /opt/libreoffice5

2.2.2 Instalación de Libreoffice

Para la utilización del procesador de textos LibreOffice Online, destinado a la edición de documentos, éste debe estar desplegado en docker. La instalación de LibreOffice Online es sencilla:

- Listado de recursos software S.O Linux
  - *docker pull libreoffice/online:master*
- Listado de recursos software S.O Windows
  - *docker run -p 9980:9980 -e 'domain=wopi\\.pruebas\\.es' libreoffice/online:master*

Es importante asegurarse de que el host donde se despliega LibreOffice Online permite la visibilidad del puerto

Además del despliegue de LibreOffice Online en docker, es necesario configurar una serie de parámetros en la Plataforma de Tramitación, los cuales se especifican más adelante.

2.3 Instalación de la Plataforma de Tramitación w@ndA

2.3.1 Configuración de los componentes w@ndA

Las configuraciones de los componentes w@ndA Port@firmas y Notific@ se realizarán sobre la instalación que se utilice de Trew@, habilitando estos componentes en cada uno de los sistemas a utilizar y estableciendo los parámetros adecuados de conexión. Esta configuración podrá ser realizada haciendo uso de la herramienta de administración de Trew@ (*TrewaAdm*).

Para más información acerca de la configuración de componentes en Trew@, se debe consultar la documentación distribuida en la última versión liberada de este producto.

2.3.2 Configuración del editor de párrafos Trew@

Para la utilización del editor de párrafos habrá que definir la constante *URL_REPORT_JAVA* con el valor: *http[s]://servidor[:puerto]/ruta-despliegue-plataforma/editor/config*. dentro de los parámetros de configuración de Trew@.

2.3.3 Despliegue en el servidor de aplicaciones Web

2.3.3.1 Configuración del servidor de aplicaciones

Será necesario contar con una instalación del contenedor de aplicaciones WildFly versión 15.0.1.Final, con el fin de desplegar en él la aplicación.

Para su instalación, existirá un usuario que será el encargado de administrar y ejecutar WildFly, y que deberá ser el propietario de la aplicación, ya que se le asignaran permisos de lectura, escritura y ejecución sobre las diversas carpetas y aplicaciones afectadas.

Por tanto, a la hora de instalar WildFly en el entorno, se creará el usuario propietario, por ejemplo “wildfly14” cuyo directorio asignado es la carpeta de instalación de WildFly, por ejemplo /usr/share/WildFly15.0.1.Final. Por motivos de seguridad es recomendable que este usuario esté autorizado para ejecutar todas las operaciones sobre la aplicación.

Será necesario obtener el paquete oficial del servidor de la página oficial de WildFly. La descarga provee un binario válido tanto para Windows como para Linux. El enlace de descarga es el siguiente:

http://download.jboss.org/wildfly/15.0.1.Final/wildfly-15.0.1.Final.zip

Una vez desplegado en el directorio correspondiente será necesario llevar a cabo unos pasos básicos para configurar los requisitos de despliegue mínimos.

- **Creación de un usuario**

  Con el fin de poder llevar a cabo la administración del servidor, es necesario que se establezca un usuario propio del servidor WildFly que tenga permisos para realizar estas acciones. Los pasos a seguir serán los siguientes:

```
// Cambiar al directorio “bin” del servidor WildFly
$ cd %WILDFLY_HOME%/wildfly-15.0.1.Final/bin/
```

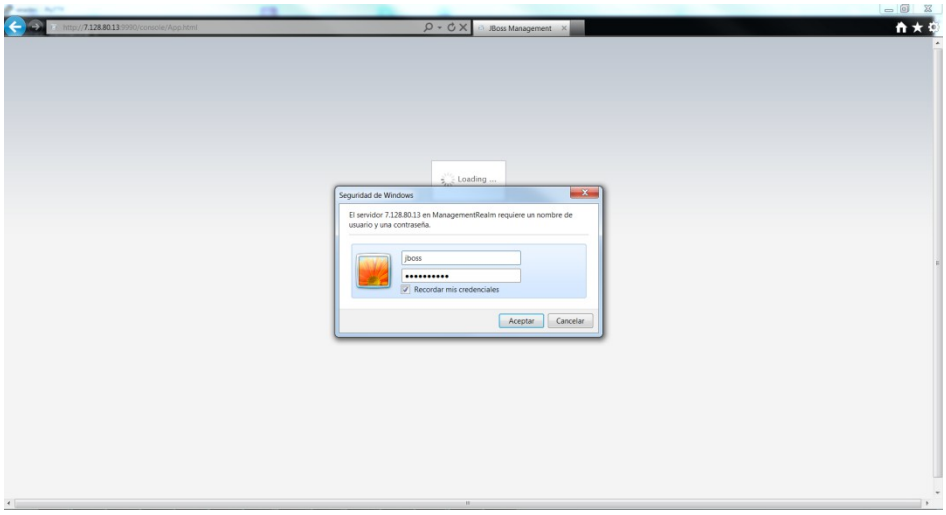
```
// Ejecutar el comando para añadir el nuevo usuario
$ ./add-user.sh
// Cuando apareca el siguiente texto indicar “a”
What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): a
// Aparecerá el siguiente mensaje para indicar el Real que se quiere utilizer. Se pulsará enter, para dejar la opción por defecto (ManagementRealm)
Enter the details of the new user to add.
Realm (ManagementRealm) :
// A continuación se pide que se indique el nombre del usuario. En este caso se indicará el que se estime oportuno.
Username : ptwanda
// El siguiente paso sera indicar la clave, teniendo en cuenta que no sea la misma que el nombre de usuario. A continuación se repetirá la clave por seguridad.
Password : ptwanda253
Re-enter Password : ptwanda253
//Indicará a qué grupos de usuario añadir el usuario a crear. Se pulsa enter sin especificar ninguno.
// Avisará del resultado de la operación y será necesario indicar “yes” para que guarde los cambios.
About to add user '<username>' for realm 'ManagementRealm'
Is this correct yes/no? yes
// A la pregunta de si utilizar el usuario creado para otras conectividades se responderá no y pulsar enter.
Is this new user going to be used for one AS process to connect to another AS process? No

// Por ultimo indica que se han actualizado correctamente los ficheros internos de WildFly.
Added user 'wildfly' to file '%WILFLY_HOME%/wildfly-15.0.1.Final/standalone/configuration/mgmt-users.properties'
Added user 'wildfly' to file '%WILFLY_HOME%/wildfly-15.0.1.Final/standalone/configuration/mgmt-users.properties'
```

- **Arranque del servidor para comprobar que todo es correcto**

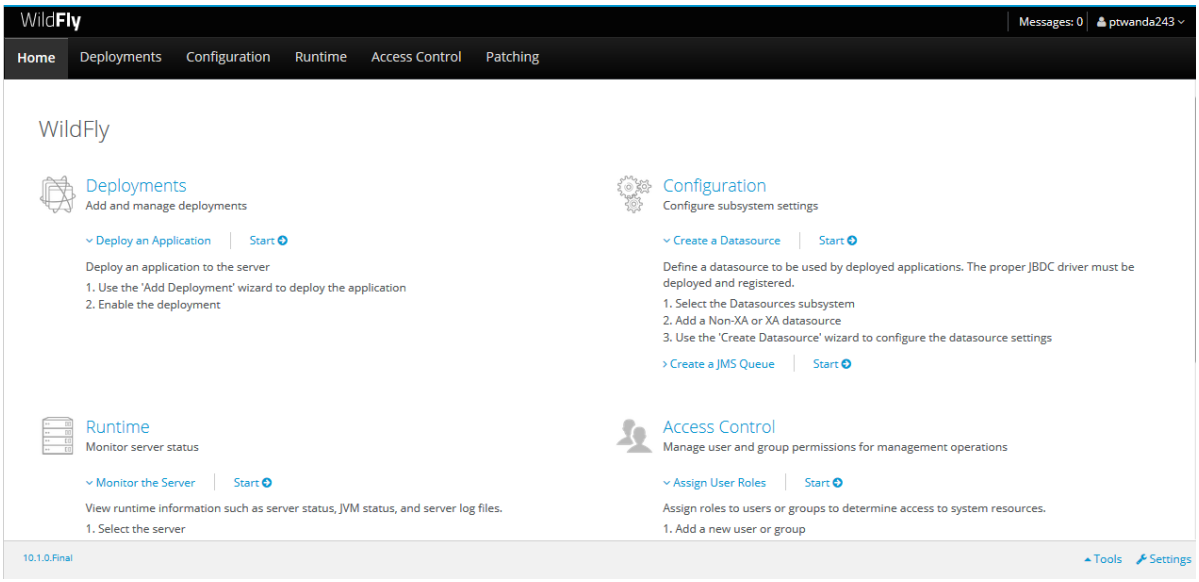
  El último paso para la configuración del servidor, será iniciarlo para comprobar que todo funciona correctamente. Para ello se indicará el siguiente comando **solo en el primer arranque,** que además permite que se pueda acceder al servidor y a la administración de forma remota, ya que se le indica la ip 0.0.0.0. Por defecto es solo accesible desde localhost (127.0.0.1).

```
$ ./standalone.sh --server-config=standalone-full.xml -Djboss.bind.address=0.0.0.0 -Djboss.bind.address.management=0.0.0.0&
```

Una vez haya arrancado sin problemas, será necesario acceder a la consola de administración indicando la URL: http://*IP_SERVIDOR*:9990/, y se deberá ver la siguiente pantalla de credenciales:



Finalmente una vez indicadas las credenciales de usuario que se ha creado, se accede a la consola de administración:



- **Configuración del modo de arranque**

  Para el correcto funcionamiento de la plataforma y del componente de indexación y búsqueda Solr es **obligatorio** arrancar el modo standalone-full del servidor de aplicaciones. Para ello es necesario editar el fichero standalone.conf, modificando el valor de la propiedad **-Djboss.server.default.config=standalone-full.xml.**

  Modificado este valor no es necesario indicar como parámetro de arranque **–server-config.**

- **Modificación de la IP de acceso público y de acceso para la administración**

  Tal y como se ha indicado en el comando para el primer arranque de WildFly, es necesario indicar la IP 0.0.0.0 para que se pueda acceder tanto a la parte pública como a la privada del servidor desde una IP distinta a localhost. Para que este cambio sea permanente y no sea necesario indicar el parámetro correspondiente en cada arranque, se llevará a cabo la configuración necesaria a través de la consola de administración.

  Para ello, en la parte superior de la consola de administración, se debe acceder al menú "*Configuration*", y en el menú que aparecerá en la parte izquierda, marcar la opción "*Interfaces*".

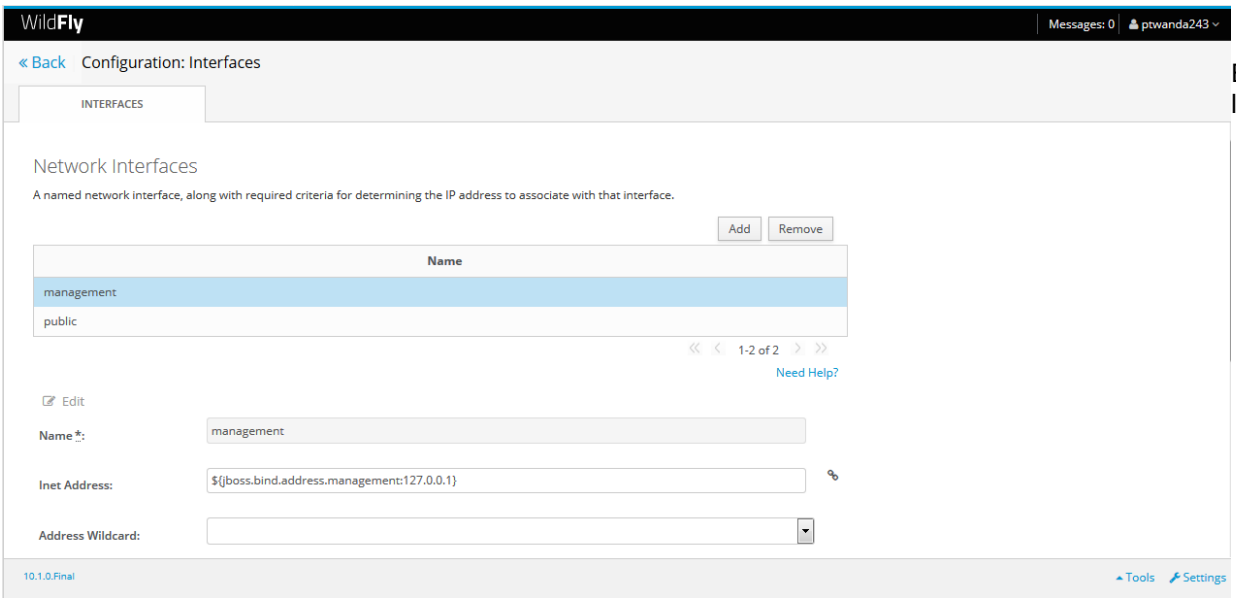
  Una vez aparezcan las interfaces, se seleccionará la de administración "*management*", y se hará clic sobre el botón editar que está justo debajo.

  Aparecerán las propiedades de la interfaz, y será necesario editar la propiedad "*Inet Address*", sustituyendo la IP de localhost 127.0.0.1, por la IP del servidor donde está alojado el JBoss, o la IP 0.0.0.0 de forma genérica.

  Una vez establecida la nueva IP, se hará clic en el botón save, que está justo encima de las propiedades.

  Para la interfaz pública "*public*", será necesario llevar a cabo el mismo procedimiento.

  En la siguiente imagen se muestra la pantalla de edición de interfaces:



En los siguientes arranques simplemente se indicará el siguiente comando:

```
$ ./standalone.sh &
```

Para parar el servidor, siempre será necesario ejecutar el siguiente comando:

```
$ ./jboss-cli.sh --connect command=:shutdown
```

- **Inclusión del módulo para la conexión con Oracle**

Se debe instalar en el servidor WildFly el módulo necesario para la conexión con el servidor de base de datos Oracle 9i, 10g ó 11G, ***ojdbc8.jar***, que se distribuye en la carpeta ***01. Software/lib***.

Para ello es necesario seguir los siguientes pasos:

1. Crear el directorio "*oracle/jdbc/main*" en el repositorio de módulos del servidor WildFly que se encuentra en la ruta *"%WildFly_HOME%/modules/system/layers/base*".
2. Incluir la librería "**ojdbc8.jar**" en el directorio creado.
3. Crear un nuevo fichero de nombre "*module.xml*" en el mismo directorio, con el siguiente contenido:

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.3" name="oracle.jdbc">
        <resources>
                <resource-root path="ojdbc8.jar" />
        </resources>
        <dependencies>
                <module name="javax.api" />
                <module name="javax.transaction.api" />
        </dependencies>

</module>
```

4. Referenciar el módulo creado en el fichero de configuración de WildFly "*standalone-full.xml*". Para ello se debe abrir el fichero *%WildFly_HOME%/standalone/configuration/standalone-full.xml* y en el subsistema "*datasources:1.0*", en el apartado "*drivers*" añadir lo siguiente:

```
<subsystem xmlns="urn:jboss:domain:datasources:4.0">
    <datasources>

                <!-- Aquí se indican todos los datasources que se necesiten -->

        <drivers>
                <driver name="h2" module="com.h2database.h2">
                <xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class>
                </driver>

                <driver name="oracle" module="oracle.jdbc">
                <xa-datasource-class>oracle.jdbc.xa.client.OracleXADataSource</xa-datasource-class>
                </driver>
        </drivers>
   </datasources>
</subsystem>
```

- **Aumento del tiempo de arranque**

Será necesario aumentar el tiempo en el que el servidor despliega las aplicaciones, para asegurar de que no se producen errores por tiempo insuficiente. Para ello se debe abrir el fichero *%WildFly_HOME %/standalone/configuration/standalone-full.xml* y en el subsistema "*deployment-scanner:1.1*", modificar lo siguiente:

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:2.0">
        <deployment-scanner path="deployments" relative-to="jboss.server.base.dir"
                scan-interval="5000"     deployment-timeout="3000"     runtime-failure-causes-rollback="${jboss.deployment.scanner.rollback.on.failure:false}" />
```

```
</subsystem>
```

- **Configuración del nodo identificador**

  Será necesario establecer un identificador único al nodo del servidor de aplicaciones. Para ello se edita el subsistema siguiente añadiendo la propiedad **node-identifier:**

```
<subsystem xmlns="urn:jboss:domain:transactions:5.0">
            <core-environment node-identifier="ptwanda253">
                <process-id>
                    <uuid/>
                </process-id>
            </core-environment>
            <recovery-environment socket-binding="txn-recovery-environment" status-socket-
binding="txn-status-manager"/>
        </subsystem>
```

- **Configuración del jvmRoute para soporte a mod_jk y mod_proxy**

  Será necesario establecer un identificador único al nodo del servidor de aplicaciones. Para ello se edita el subsistema siguiente añadiendo la propiedad **instance-id:**

```
<subsystem xmlns="urn:jboss:domain:undertow:8.0" instance-id="ptwanda253">
```

- **Aumentar número de parámetros peticiones HTTP/HTTPS**

  Será necesario ampliar el número máximo de parámetros en peticiones HTTP/HTTPS. Para ello se edita el subsistema siguiente añadiendo la propiedad **max-parameters:**

```
<subsystem xmlns="urn:jboss:domain:undertow:8.0" instance-id="ptwanda253">
 <buffer-cache name="default"/>
 <server name="default-server">
 <http-listener name="default" socket-binding="http" max-parameters="20000" redirect-
socket="https" enable-http2="true"/>
 <https-listener name="https" socket-binding="https" max-parameters="20000" security-
realm="ApplicationRealm" enable-http2="true"/>
```

- **Aumentar tiempo de timeout de cachés infinispam.**

  Para ello será necesario editar el siguiente subsistema definiendo la propiedad **acquire-timeout**:

```
<subsystem xmlns="urn:jboss:domain:infinispan:7.0">
        <cache-container name="server" default-cache="default"
module="org.wildfly.clustering.server">
            <local-cache name="default">
               <transaction mode="BATCH"/>
            </local-cache>
        </cache-container>
        <cache-container name="web" default-cache="passivation"
module="org.wildfly.clustering.web.infinispan">
            <local-cache name="passivation">
               <locking acquire-timeout="1200000" isolation="REPEATABLE_READ"/>
               <transaction mode="BATCH"/>
               <file-store passivation="true" purge="false"/>
            </local-cache>
            <local-cache name="persistent">
               <locking acquire-timeout="1200000" isolation="REPEATABLE_READ"/>
               <transaction mode="BATCH"/>
               <file-store passivation="false" purge="false"/>
```

```
            </local-cache>
            <local-cache name="concurrent">
               <file-store passivation="true" purge="false"/>
            </local-cache>
         </cache-container>
         <cache-container name="ejb" aliases="sfsb" default-cache="passivation"
module="org.wildfly.clustering.ejb.infinispan">
            <local-cache name="passivation">
               <locking acquire-timeout="1200000" isolation="REPEATABLE_READ"/>
               <transaction mode="BATCH"/>
               <file-store passivation="true" purge="false"/>
            </local-cache>
            <local-cache name="persistent">
               <locking acquire-timeout="1200000" isolation="REPEATABLE_READ"/>
               <transaction mode="BATCH"/>
               <file-store passivation="false" purge="false"/>
            </local-cache>
         </cache-container>
         <cache-container name="hibernate" default-cache="local-query"
module="org.hibernate.infinispan">
            <local-cache name="entity">
               <transaction mode="NON_XA"/>
               <eviction strategy="LRU" max-entries="10000"/>
               <expiration max-idle="100000"/>
            </local-cache>
            <local-cache name="local-query">
               <eviction strategy="LRU" max-entries="10000"/>
               <expiration max-idle="100000"/>
            </local-cache>
            <local-cache name="timestamps"/>
         </cache-container>
      </subsystem>
```

- **Modificación del módulo org.jdom**

  Para el correcto funcionamiento de la plataforma es necesario incluir una dependencia en el módulo org.jdom. Para ello se accede **modules\system\layers\base\org\jdom\main** y se edita el fichero module.xml, añadiendo la siguiente dependencia dentro de la etiqueta <dependencies>: **<module name="org.apache.xerces" />.**

- **Instalación del módulo org.bouncycastle**

  Es necesario incluir un nuevo módulo de WildFly **bouncycastle**. Para ello hay que descomprimir el fichero **bouncycastle.zip** ubicado en la ruta /**04.Recursos desarrollo/ficheros** del entregable entregable de la plataforma en la ruta "/usr/local/WildFly-15.0.1.Final-ptwanda/modules/*system/layers/base*/org". Después de descomprimir debe existir la ruta "/usr/local/WildFly-15.0.1.Final-ptwanda /modules/*system/layers/base*/org/bouncycastle/main".

- **Instalación del módulo com.sum.jsf-Impl**

  Es necesario incluir una versión específica de jsf-impl para la compatibilidad con el componente Formul@. Para ello, copiar los ficheros ubicados en /04 Recursos desarrollo/ficheros/jsf-impl.zip en la ruta "[WildFly15.0.1Final]/modules/system/layers/base/com/sun/jsf-impl" sobrescribiendo el fichero module.xml.

2.3.3.2 Instalación de la aplicación

La plataforma de tramitación se distribuye en formato de fichero WAR (Web Application Archive).

El despliegue de la aplicación PTw@ndA debe realizarse en un directorio que contenga el nombre de la aplicación con la extensión ".war". Ej: ptwanda-web.war

Para ello, el fichero WAR distribuido en el entregable de la aplicación, debe descomprimirse en un directorio situado en la ruta `{JBOSS}/standalone/deployments` y eliminarse de dicho directorio.

De esta manera, no existirá desplegado un fichero .war con la aplicación, sino únicamente el directorio descomprimido con la extensión .war

Es necesario crear un fichero denominado **ptwanda-web.war.dodeploy,** donde el nombre ptwanda-web.war debe coincidir con el nombre del directorio donde se ha descomprimido el war. Una vez creado se edita y se escribe el nombre del directorio anterior, en este caso, **ptwanda-web.war**. Este paso es **obligatorio** para indicarle a WildFly que realice el despliegue de la aplicación que se encuentra dentro del directorio con extensión .war.

2.3.3.3 Configuración de conexiones con Base de datos

Una vez realizado el despliegue de la aplicación en el servidor de aplicaciones, se debe proceder a configurar la aplicación para su adecuado funcionamiento. Todas las conexiones a BBDD se realizan a través de fuentes de datos (DataSources) que serán definidos en el servidor de aplicaciones WildFly.

- **Conexión a la Base de datos de Trew@:** la conexión con la base de datos de Trew@ se realizará mediante un pool de conexiones gestionado por el propio servidor de aplicaciones. Para la conexión con Trew@ será necesario la creación de dos datasources, uno denominado por defecto TrewaDS (este valor puede cambiarse, configurando el nuevo valor en la tabla CONFIG de PTw@ndA) y otro denominado default.

Para el caso de utilizar soporte Multi-Trew@ (conexión con varias instancias de Trew@) será necesario definir el mismo prefijo para cada jndi correspondiente a cada base de datos de Trew@, siendo obligatorio utilizar el prefijo Trew@ en la declaración del jndi. Por ejemplo, para una instancia de Trew@ para el organismo Consejería de Hacienda y Administración Pública se tendría la siguiente definición del datasource:

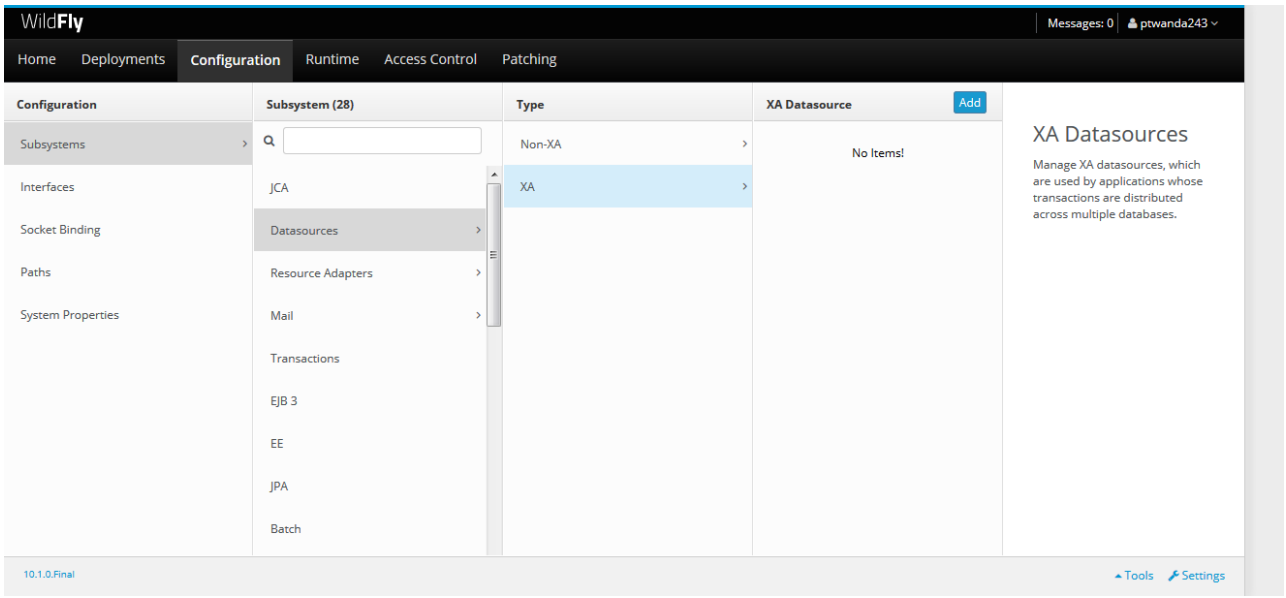
```
<datasource jta="false" jndi-name="java:/TrewaCHAP" pool-name="TrewaCHAP" enabled="true" use-ccm="false">
```

Con la configuración multi-Trew@ activada será necesario seguir definiendo un dataSource por defecto TrewaDS, que será el configurado en el parámetro de configuración TREWACONEXION. Esta instancia de Trew@ será la utilizada para el proceso de login en la administración para el usuario super-administrador que tendrá permisos sobre todas las instancias del motor de tramitación Trew@.

El nombre del jndi definido (TrewaCHAP) deberá utilizarse a la hora de crear el core de indexación asociado a la instancia de Trew@, empleando el mismo nombre en minúsculas. Por ejemplo, para un jndi denominado **TrewaCHAP** el nombre del core de Solr asociado deberá ser **trewachap**.

Los datasources se darán de alta desde la herramienta de administración de WildFly.

Una vez se haya accedido a la consola de administración, en la parte superior, se debe acceder al menú "*Configuration*", y en el menú que aparecerá en la parte izquierda, acceder a Subsystems – DataSources – XA. En el panel que aparece en la parte derecha, encima del listado de datasources definidos, aparece el botón "*Add*", en el cual habrá que pulsar para dar de alta un nuevo datasource. En la siguiente imagen se muestra el proceso:

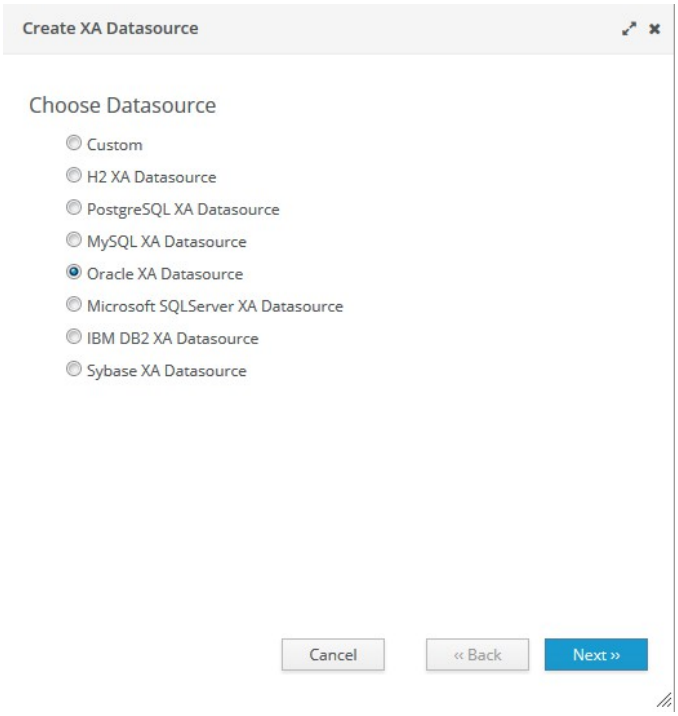


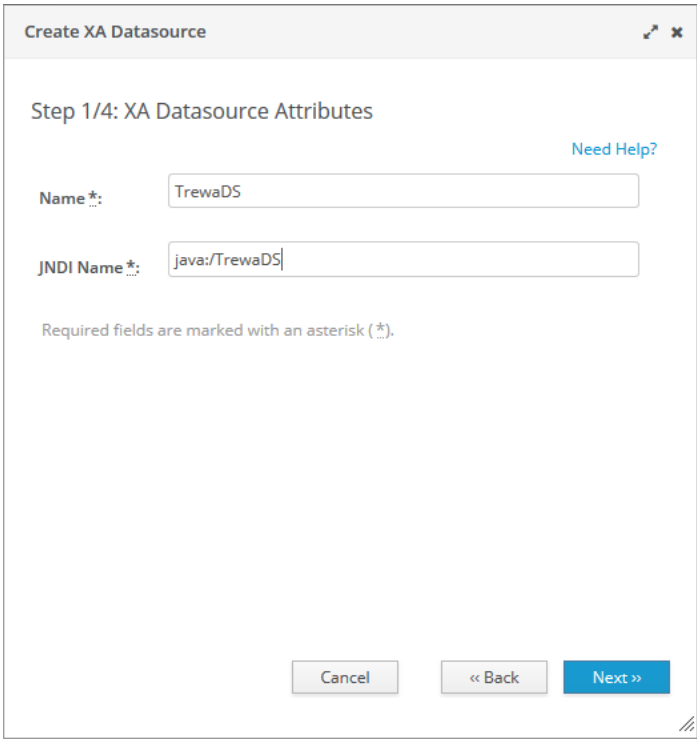
Una vez aparezca el asistente para crear un nuevo datasource, indicar lo siguiente para cada una de las ventanas que aparecen:

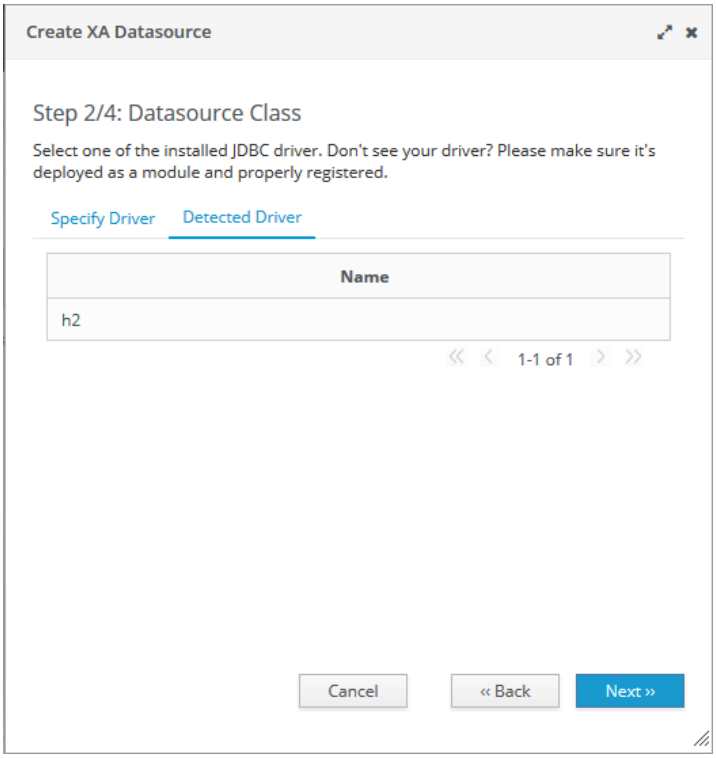
- Primera pantalla:
  - Elegir DS Oracle.
- Primera pantalla:
  - **Name:** TrewaDS
  - **JNDI Name:** java:/TrewaDS
- Segunda pantalla:
  - Elegir el driver de oracle que se ha creado en el punto 3.3.3.2.1.
- Tercera pantalla:
  - **URL:** jdbc:oracle:thin:@IP_BBDD:PUERTO_BBDD:NOMBRE_ESQUEMA
- Cuarta pantalla:
  - **Username:** USUARIO_DEL_ESQUEMA
  - **Password:** CLAVE_DEL_USUARIO

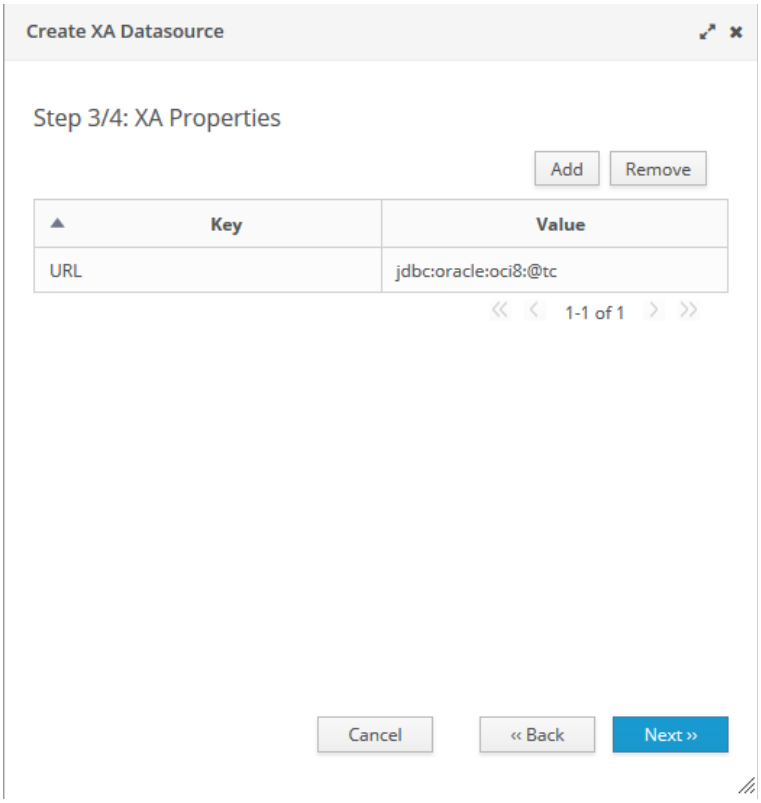
Una vez completado los campos, se hará clic en el botón "*Done*" y quedará guardado el datasource.

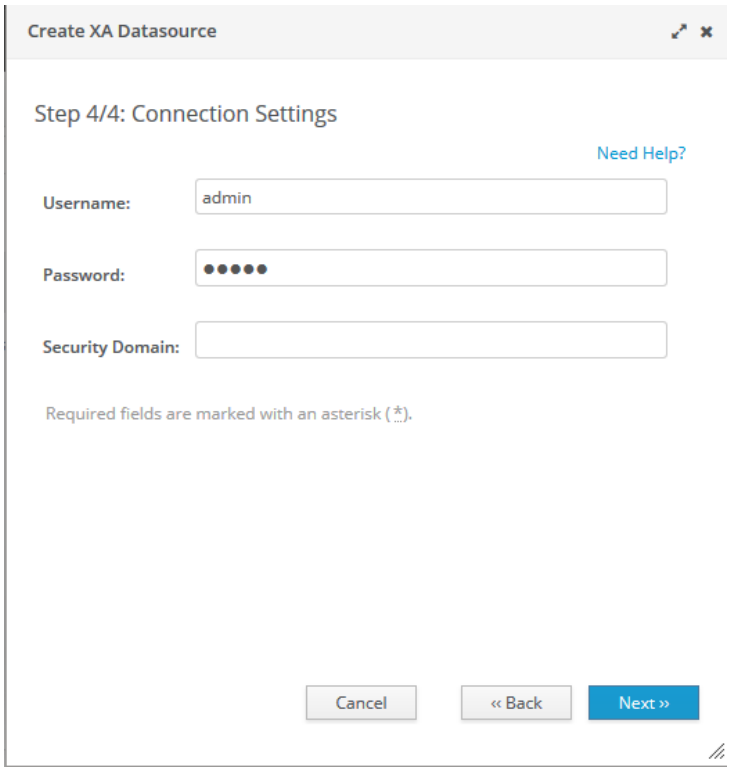
A continuación se muestra en imágenes el proceso explicado:






Create XA Datasource
Step 2/4: Datasource Class
Select one of the installed JDBC driver. Don't see your driver? Please make sure it's deployed as a module and properly registered.
Specify Driver
Detected Driver
Name
h2
1-1 of 1
Cancel
« Back
Next »


Create XA Datasource
Step 3/4: XA Properties
Add
Remove
Key
Value
URL
jdbc:oracle:oci8:@tc
1-1 of 1
Cancel
« Back
Next »



En este momento, ha quedado configurado el datasource con la configuración por defecto. Sin embargo, es necesario configurar manualmente algunas propiedades con otros valores optimizados. Para ello, será necesario seleccionar en el listado de datasources disponibles el que se acaba de crear, y cambiar los valores que se indican a continuación en cada una de las pestañas.

- Statement
  - Statement Cache Size: 32
- Timeouts
  - blocking-timeout-millis: 5000
  - idle-timeout-minutes: 30
- Pool
  - Min Pool Size: 10
  - Max Pool Size: 100

Finalizada la configuración de TrewaDS será necesario repetir los pasos anteriores para crear el datasource **default.**

- **Conexión con la Base de datos de Plataforma:** la conexión con el modelo de datos propio de PTw@ndA se realizará configurando en el archivo *hibernate.properties* una referencia a un DataSource declarado en el servidor de aplicaciones. El usuario de BBDD vinculado a este DataSource deberá tener habilitados los permisos para crear tablas y para operar sobre éstas (insert, update, select y delete).

El archivo *hibernate.properties* debe quedar configurado tal y como se indica a continuación:

```
## HIBERNATE. Configuración de acceso al esquema propio de plataforma.

#Datasource de datos paramétricos de Plataforma (configurado en server.xml)
hibernate.connection.jndi=Plataforma
hibernate.connection.release_mode=auto
hibernate.dialect=org.hibernate.dialect.OracleDialect
hibernate.hbm2ddl.auto=update
hibernate.show_sql=false
transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory
java.naming.factory.initial=org.jnp.interfaces.NamingContextFactory

#Datasource de datos no-paramétricos de Plataforma (configurado en server.xml)
hibernate.connection.jndi.datos=PlataformaDatos
hibernate.connection.release_mode.datos=auto
hibernate.dialect.datos=org.hibernate.dialect.OracleDialect
hibernate.hbm2ddl.auto.datos=update
hibernate.show_sql.datos=false
transaction.factory_class.datos=org.hibernate.transaction.JDBCTransactionFactory
java.naming.factory.initial.datos=org.jnp.interfaces.NamingContextFactory
```

No es necesario modificar este fichero a menos que se desee emplear un nombre JNDI distinto al valor por defecto para el Datasource, esto es, "*Plataforma o PlataformaDatos"*.

Será necesario crear los dos datasources, **Plataforma** y **PlataformaDatos** siguiendo los mismos pasos descritos anteriormente para el datasource **TrewaDS**..

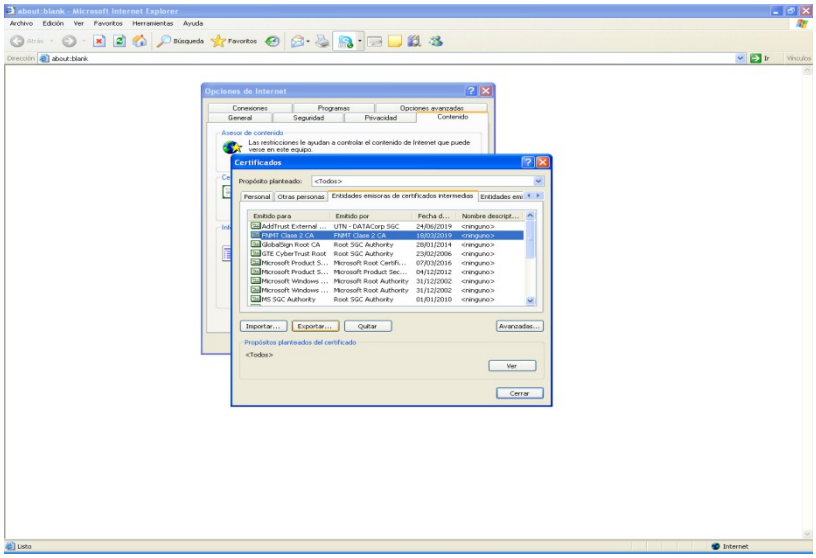
2.3.4 Almacén de certificados

Para poder establecer la comunicación SSL entre el servidor de aplicaciones de la Plataforma de Tramitación y el servidor de custodia de @firma, se debe importar la clave pública SSL de la FNMT (Fábrica Nacional de Moneda y Timbre) dentro del almacén de certificados de confianza de la JVM con la que se ejecuta el servidor de aplicaciones de PT.

Para ello, la forma más sencilla es obtener el certificado con la clave pública de FNMT desde el navegador web Internet Explorer de un microordenador que ya disponga de él, exportarlo y, seguidamente, importarlo desde el servidor de aplicaciones.

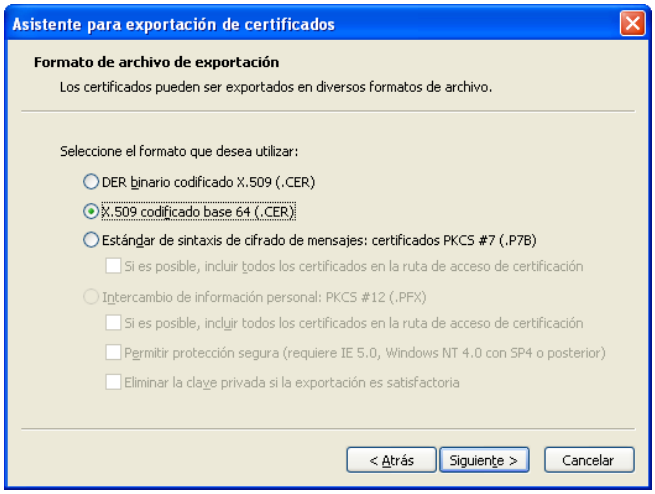
Para exportarlo, seguiremos el siguiente recorrido en los menús y opciones del navegador: Herramientas → Opciones de Internet... → Contenido → Certificados... → Entidades Emisoras Raíz de Confianza

Llegamos a la siguiente pantalla:

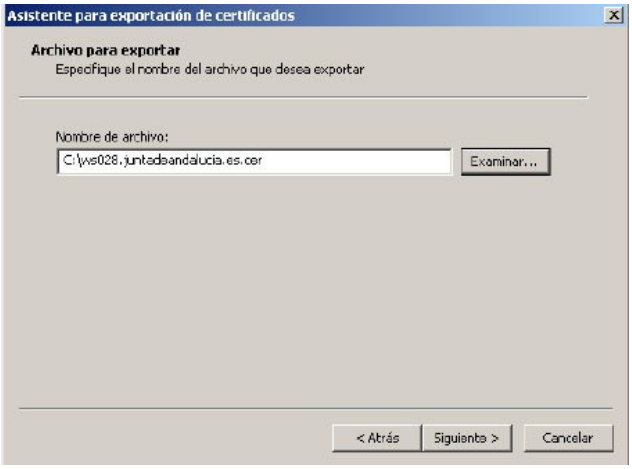


Marcaremos el certificado de la FNMT y pulsamos sobre “Exportar”:

Pulsar sobre “Siguiente” y seleccionar “X.509 codificado base 64”:



A continuación, nos pedirá la ruta donde queremos guardarlo.



Tras esto, ya tenemos el certificado público SSL de la FNMT, que podremos importar mediante la herramienta "keytool" de java sobre el almacén por defecto, "cacerts" (la clave de este almacén es "changeit"), del runtime de java, localizado en el siguiente directorio:

```
${JAVA_HOME}/jre/lib/security
```

Para incluirlo, deberemos importarlo mediante el siguiente comando, que siguiendo con el ejemplo sería:

```
keytool -import -keystore ${JAVA_HOME}/jre/lib/security/cacerts -file <ruta-fichero-fnmt> -alias FNMT
```

Tras realizar la importación de la parte pública es necesario incluir como parámetros de la máquina virtual de Java el almacén de certificado de confianza.

Desde la consola de administración de WildFly se accede a la opción *System Properties* del menú *Configuration* y añadimos las siguientes dos propiedades:

javax.net.ssl.trustStore = `${JAVA_HOME}/jre/lib/security/cacerts`

javax.net.ssl.trustStorePassword = *changeit*

2.3.5 Configuración de conexión SSL

Para utilizar servicios de otros componentes (SCSP, @ries, .,..) publicados bajo el protocolo HTTPS, es necesario incluir como parámetros de la máquina virtual de Java un almacén de certificados de confianza propio junto con la contraseña de acceso.

Desde la consola de administración de WildFly se accede a la opción *System Properties* del menú *Configuration* y añadimos las siguientes dos propiedades:

javax.net.ssl.keyStore = `${JAVA_HOME}/jre/lib/security/cacerts`

javax.net.ssl.keyStorePassword = *changeit*

La clave privada con la que se importen los certificados debe coincidir con la clave que posee el propio almacén de certificados.

Es importante que todos los certificados clientes que se vayan a usar estén en el mismo almacén de certificados, la JVM es quien selecciona el certificado apropiado para la conexión a establecer sin que sea necesario dar un alias del certificado dentro del almacén de certificados.

En caso de que la aplicación se encuentre en modo clúster, esta operación debe realizarse en todos y cada uno de los servidores que componen el clúster.

2.3.6 Ampliación de memoria de la Máquina Virtual Java en WildFly

En el caso de realizar una instalación por defecto de WildFly, se estará ejecutando la máquina virtual Java con un tamaño de memoria de 64 megas. Este límite pudiera resultar insuficiente para el correcto funcionamiento de PTw@ndA y resto de aplicaciones web desplegadas en el mismo.

Esto puede detectarse mediante la aparición de errores del tipo `java.lang.OutOfMemoryError` en la consola del servidor, unido a fallos en la ejecución de las aplicaciones instaladas. A fin de evitar dicho error, se habrá de añadir a la variable de sistema JAVA_OPTS un valor máximo de memoria de 256 MBytes.

Si se siguieran apreciando caídas (porque exista elevado número de aplicaciones desplegadas en el mismo servidor de aplicaciones o sean grandes consumidores de memoria), la recomendación obvia es seguir ampliando el valor de dicho parámetro en función a las características del servidor y recursos disponibles.

A continuación se explica cómo modificar este parámetro:

Es necesario acceder al fichero `standalone.conf` ubicado en la ruta `{WildFly}/bin` y editarlo, localizando para ello la asignación que se realiza a la variable **JAVA_OPTS**:

```
#
# Specify options to pass to the Java VM.
#
if [ "x$JAVA_OPTS" = "x" ]; then
           JAVA_OPTS="-Xms128m    -Xmx512m    -XX:MaxPermSize=256m    -Djava.net.preferIPv4Stack=true
-Dorg.jboss.resolver.warning=true                              -Dsun.rmi.dgc.client.gcInterval=3600000
-Dsun.rmi.dgc.server.gcInterval=3600000"
                  JAVA_OPTS="$JAVA_OPTS      -Djboss.modules.system.pkgs=$JBOSS_MODULES_SYSTEM_PKGS
-Djava.awt.headless=true"
   JAVA_OPTS="$JAVA_OPTS -Djboss.server.default.config=standalone.xml"
else
   echo "JAVA_OPTS already set in environment; overriding default settings with values: $JAVA_OPTS"
fi
```

La próxima vez que se inicie el servidor, este arrancará con la configuración descrita anteriormente, habiendo resultado aumentada la memoria a los tamaños asignados.

Los valores especificados anteriormente dependerán de la memoria que disponga el servidor en el que se encuentra ubicado WildFly, y del rendimiento y carga que vaya a tener la aplicación en el entorno que se despliegue: desarrollo, pruebas, formación, producción,… Para entornos de producción pueden especificarse valores de memoria mayores: 521, 1024, 2048, …

2.3.7 Configuración de la conexión a Internet del servidor

PTw@ndA requiere conexión a Internet debido a que el componente *struts* necesita descargar la definición sintáctica DTD de sus ficheros XML de configuración.

En el caso de que el servidor disponga de conexión directa a http://struts.apache.org, no se requiere hacer ninguna tarea adicional de configuración.

Para el caso contrario, en el que no exista una conexión directa sino a través de servidor proxy, será necesario establecer la configuración del mismo en nuestra JVM, lo cual se consigue definiendo unos parámetros en la variable JAVA_OPTS.

Desde la consola de administración de WildFly se accede a la opción *System Properties* del menú *Configuration* y añadimos las siguientes propiedades:

```
http.proxyHost=servidorProxy
http.proxyPort=puertoProxy
http.proxyUser=usuarioProxy           [si es necesario]
http.proxyPassword=claveProxy         [si es necesario]
http.nonProxyHosts=Filtro de direcciones locales.
```

2.3.8 Configuración de la Plataforma de Tramitación w@ndA

2.3.8.1 Ejecución de scripts de BBDD

Como alternativa al punto anterior, a partir de esta versión de PTw@ndA, se proporcionan unos scripts de BBDD que permiten la creación del esquema, usuarios, tablas y datos paramétricos, sin necesidad de realizar el paso anterior.

Los pasos a realizar son los siguientes:

i. Acceder con usuario system/sys y ejecutar los script ubicados en el directorio *02. Scripts BBDD* \ (previa modificación de las localizaciones de los tablespaces).
`01.Tablespaces.sql`
`02.Usuarios.sql` (previa modificación de las constantes con los nombres de los tablespaces)

ii. Acceder con usuario PTW_OWNER y ejecutar los scripts siguientes en el orden dado (previa modificación de las constantes con los nombres de los tablespaces creados):
`03.Tablas_PTW_OWNER.sql`

```
06.Secuencia_PTW_OWNER.sql
```

iii. Acceder con usuario PTW_TPOWNER y ejecutar los scripts siguientes en el orden dado:
```
04.Tablas_PTW_TPOWNER.sql
```
(previa modificación de las constantes con los nombres de los tablespaces)
```
05.Datos_PTW_TPOWNER.sql
07.Secuencia_PTW_TPOWNER.sql
```

iv. Acceder con usuario system/sys y ejecutar el script (previa modificación de las constantes con los nombres de los tablespaces creados):
```
08.Usuarios_Trabajo.sql
```

v. Acceder con usuario PTW_TPUSER y ejecutar el script
```
09.Sinonimos_PTW_TPUSER.sql
```

vi. Acceder con usuario PTW_USER y ejecutar el script
```
10.Sinonimos_PTW_USER.sql
```

vii. Modificar las propiedades relativas a la creación o sincronización de hibernate con el esquema de BBDD; para ello es necesario cambiar el valor de las siguientes propiedades del fichero `hibernate.properties`:

```
hibernate.hbm2ddl.auto=false
hibernate.hbm2ddl.auto.datos=false
```

Una vez realizado este punto, se configurará la aplicación a partir de lo descrito en el siguiente apartado, y al arrancar el servidor estará la aplicación PTw@ndA correctamente desplegada y configurada.

2.3.8.2 Configuración de la aplicación

La configuración de PTw@ndA está centralizada en la tabla CONFIG del esquema propio de PTw@ndA (PlataformaDatos).

En la tabla se detallan parámetros de necesaria modificación en toda implantación de PTw@ndA (tipo I). Existen otros parámetros que pueden requerir, ocasionalmente, un cambio (tipo O) y aquellos otros que raramente deban cambiarse (tipo E). En esta tabla sólo aparecen los parámetros que son necesarios tener configurados para arrancar la aplicación inicialmente, existen otros parámetros que al no ser necesaria su configuración para este primer arranque, se ha trasladado su configuración al manual de Administración, dónde se explica de forma detallada la forma de realizarse.

Tipo	Parámetro	Descripción	Ejemplo	Obligatorio
Propiedades de Trew@				
I	TREWASISTEMA	Sistema Trew@ utilizado por defecto	TREW@	SI
I	TREWASISTEMAS	Diferentes Sistemas que pueda usar PTw@ndA. Se definen separados por comas.	TREW@,CJAP	SI

I	TREWACONEXION	DataSource de conexion a la base de datos de Trew@	TrewaDS	SI
login/@firma				
O	MODO_LOGIN	Indica la forma de logarse por defecto (2 = Autenticación con DNI,| 1 = Auntenticación con @firma | 3 = Autenticación vía LDAP)	2	SI
I	SERVIDOR_FIRMA	IP o nombre DNS del servidor @firma	wsXXX.juntadeandalucia.es	SI
port@firmas				
I	ID_APLICACION_PORTA_FIRMA	Aplicación a utilizar en port@firma para el alta y consulta de peticiones.	idPlataformaEnPortafirmas	NO
I	URL_PORTA_FIRMA	Acceso al informe de firma. Servidor port@firmas.	http[s]://servidor-portafirma[:puerto]/pfirma/services/PfServicioWS	SI
I	CONVERSION_PDF_PORTAFIRMAS	Indica si hay que convertir a pdf los documentos antes de mandarlos a firmar.	true	SI
I	PORTAFIRMAS_ENVIO_CORREO_AVISO	Indica si se notificará por vía e-mail el envío de documentos a firmar.	false	NO
Conversión de documentos a PDF con OpenOffice				
I	IP_OPENOFFICE	Servidor donde encontramos el servidor OpenOffice (solo es posible indicar localhost)	localhost	SI
I	PUERTO_OPENOFFICE	Puerto donde debe escucha el servidor OpenOffice.	2002	SI
Sincronización				
I	APLICACION_SINCRONISMO_TRAMITADOR	Si es “true”, el sistema sincronizará la información en el esquema de datos propio de la plataforma con la información del sistema o sistemas Trew@ de forma automática tras el despliegue de la aplicación.	true	SI
I	APLICACION_CREACION_INSTALACIONES_DEFECTO	Si es “true”, el sistema creará la instalación “DEFECTO” de forma automática tras el despliegue de la aplicación.	true	SI
Documento				
O	DOCUMENTO_TIPO_INCORPORAD	Nombre del tipo de documento de	NoDefinido	SI

	O_NO_DEFINIDO	Trew@ que se utilizará para incorporar documentos no definidos en los flujos de tramitación, por ejemplo en la utilidad de incorporación de documentos.		
O	DOCUMENTO_TIPO_INCORPORADO_NO_DEFINIDO_INFORMABLE	Nombre del tipo de documento de Trew@ que se utilizará para incorporar documentos no definidos en los flujos de tramitación, por ejemplo en la utilidad de incorporación de documentos y a que además son informables.	NoDefInf	SI
O	DOCUMENTO_SUSTITUIRFIRMADOS	Para insertar los documentos firmados en trewa con cajetín		SI
Perfil				
E	ADMINISTRACION_PERFIL	Perfil mínimo que debe tener un usuario para hacer uso de la herramienta de administración.	TR_R_USUARIO	SI
E	ESCRITORIO_PERFIL	Perfil mínimo que debe tener un usuario para hacer uso del escritorio.	TR_R_USUARIO	SI
I	USUARIO_SERVICIO_WEB	Código de usuario del tramitador definido en trew@ que se utilizará para realizar las operaciones desde los servicios WEB.	PL_WEBUSR	SI
Otras				
I	DEBUG	Si es “true”, habilita el servidor en modo debug.	false	SI
I	URL_ESCRITORIO	URL DE ACCESO AL ESCRITORIO	http[s]://servidor[:puerto]/ruta-despliegue-plataforma	SI
E	CONEXION_TRAMITADOR_FIJA	Si es “true”, habilita las conexiones fijas Trew@ para el escritorio de tramitación. Indicar solo “true” o “false”.	false	SI
E	CONEXION_ADMINISTRACION_FIJA	Si es “true”, habilita las conexiones fijas Trew@ para la herramienta de administración. Indicar solo “true” o “false”.	false	SI
I	CACHEO_API_NUMERO	Es el número máximo de API`s asignadas a un solo usuario.	1	SI
I	CACHEO_API	Indica si queremos que se haga un cacheado de la API de TREW@	true	SI

I	DESCARGAR_MODULOS	Indica si queremos instalar los módulos verticales almacenados en BBDD,	true	SI
I	VISUALIZAR_PANTALLA_INTERESADOS_EXPEDIENTE	Si es “false”, deshabilita la pantalla de asociar interesados en el alta de un expediente.	true	SI
I	EVITAR_FILTROS_FASE_EXPEDIENTE	Si es “true”, no se diferencia entre diferentes instancias de la misma fase para el cálculo de documentos y tareas.	false	SI
I	VERSION_PLATAFORMA	Versión de la plataforma	V2.5.7	SI
I	RUTA_DOCUMENTO_FAQ	Ruta donde se encuentra el documento de FAQ.	http://localhost:8080/ptwanda-web/modulos/faq/PTW230_FAQ_Preguntas_y_respuestas_frecuentes_Tramitador_v01r00.pdf	SI
I	PROCEDIMIENTO_OBLIGATORIO_BUSQUEDA	Si es “true”, es necesario especificar el procedimiento a la hora de realizar las búsquedas, genérica o expandida.	true	SI
I	ACTIVO_SOLR_CLOUD	Indica si la plataforma está haciendo uso de SolrCloud, en el caso de estar configurado en clúster. Si la plataforma se despliega sobre un único nodo, el parámetro debe tener valor false.	false	SI
I	ZOOKEEPER_ENDPOINTS	URLs de servidores zookeeper que contiene los ficheros de configuración de Solr, cuando la plataforma está desplegada en clúster.	servidor1.2181,servidor2:2181,servidor3:2181	SI
I	DELAY_RECARGA_PARAMETROS_CONFIG	Intervalo de tiempo, expresado en milisegundos, entre cada ejecución de recarga de parámetros de configuración.	300000	SI
I	TIPO_ESCRITORIO	Indica el escritorio de tramitación configurado: diseño basado en portlets(PORTLET) o diseño basado en pestañas(PESTANA).	PESTANA	NO
I	ACTIVAR_RECARGA_PARCIAL_FORMULA	Indica si se desea activar la funcionalidad de recarga parciales en los formularios del motor de formularios Formul@, disponible a partir de la versión 4.0.0.	true	NO

I	TIPO_UTILIDADES	Indica, para el escritorio configurado por pestaña, si de desea mostrar las utilidades como pestaña dentro del acordeón (PESTANA) o bien configurando todas las utilidades visibles en la zona de información general del expediente (PORTLET).	PESTANA	NO
I	NUMERO_HILOS_MAXIMO	Número máximo de hilos creados en los procesos de generación masiva de documentos. (Configurar un valor demasiado elevado puede causar un aumento del consumo de memoria)	20	NO
I	ASUNTO_PFIRMA	Campo asunto enviado a Port@firmas cuando se pone a disposición del firmante el documento a firmar, pudiéndose definir variables que serán sustituidas antes de realizar la operación de envío del documento a firmar.	El documento de nombre $$DOCUMENTO_PFIRMA$$ del expediente $$EXPEDIENTE_PFIRMA$$ enviado a firmar en la fase $$FASE_PFIRMA$$ tiene como interesado/s a $$INTERESADOS_PFIRMA$$ y como firmante/s a $$FIRMANTES_PFIRMA$$.	
I	EDITOR_TEXTOS_DEFECTO	Indica el editor de textos a utilizar para la edición de documentos: WEBOFFICE ó EDITOR_WEB	WEBOFFICE	NO
I	CONFIGURACION_MULTITREWA_ACTIVADO	Indica si la plataforma opera con más de una instalación del motor de tramitación Trewa.	false	NO
I	WOPI_ENDPOINT	Indica la URL donde se encuentra el componente WOPI	http://<ip>:<port>	NO
I	LIBREOFFICE_ENDPOINT	Indica la URL donde se encuentra desplegado el componente LibreOffice Online	http://<ip>:<port>/loleaflet/dist/loleaflet.html	NO
I	MIN_CADUCIDAD_LIBREOFFICE	Indica el tiempo de caducidad de la sesión en LibreOffice Online	30	NO
I	MODO_VISUALIZACION_LIBREOFFICE	Indica el modo de visualización de LibreOffice Online	PESTANA / VENTANA	NO
I	MOSTRAR_INFO_TRANSICIONES	Indica si la información para la ayuda a la tramitación se encuentra activada o desactivada	Fasle	SI

2.3.8.3 Configuración del motor de indexación y búsqueda

El motor de indexación y búsqueda de la Plataforma de Tramitación w@ndA está compuesto en su núcleo por el proyecto **Solr de Apache Lucene**. Las características de dicho sistema son:

- Avanzadas capacidades de búsqueda *full-text*.
- Optimizado para soportar un volumen de tráfico elevado.
- Basado en interfaces abiertas, como XML y http.
- Escalable.
- Flexible y parametrizable en base a archivos de configuración en formato XML.
- Arquitectura extensible en base a *plug-ins*.

Este componente de la Plataforma de Tramitación habilitará búsqueda sobre el motor de tramitación Trew@.

A continuación se detalla la configuración del motor de indexación Solr, distinguiéndose si PTw@ndA está configurada en modo clúster:

2.3.8.3.1 Configuración Solr para un único nodo.

Si la plataforma de tramitación w@ndA se encuentra desplegada en un único nodo, no es necesario configurar el componente Solr en clúster.

A continuación se describen los pasos a seguir para la correcta configuración del componente Solr:

1. Copiar la carpeta solr-8.0.0 existente en la ruta /05.Software base/Distribución_Solr/01 Solr_Mono_Trewa/ del entregable de PTw@ndA al directorio donde se encuentra el servidor WildFly.

   En el caso de integrarse con más de una instancia del motor de tramitación Trew@ el paso 2 debe omitirse y ejecutar las instrucciones del punto Soporte Multi-Trew@ para un único nodo Soporte Multi-Trew@ para un único nodo.

2. Incluir el parámetro de arranque de WildFly para indicar la ruta al directorio de solr copiado en el paso 1, desde la administración de WildFly o directamente en el fichero de configuración standalone. Para el caso de haber copiado el directorio dentro del /opt sería:

   solr.solr.home=/opt/solr-8.0.0/server/solr

3. Creación de los siguientes parámetros de configuración desde la Administración de PTw@ndA:

   a. ACTIVO_SOLR_CLOUD: Parámetro que indica si se ha realizado la configuración del componente SolrCloud. Para este caso, el valor a establecer será false.

   b. SOLR_ENDPOINTS: URL en la que se encuentra solr: servidor:8983

Para una configuración avanzada del motor de búsqueda se requiere la edición de los archivos de configuración propios de Solr. Se insiste nuevamente en que la configuración por defecto será válida para la mayoría de las instalaciones de PT:

schema.xml

solrconfig.xml

2.3.8.3.1.1 Soporte Multi-Trew@ para un único nodo

En el caso de desplegar la plataforma de tramitación para sincronizarse con más de una instancia del motor de tramitación Trew@ se deberán realizar los siguientes pasos:

1. Copiar la carpeta **solr-8.0.0** existente en la ruta **/05.Software base/Distribución_Solr/ 00. Soporte_Multi_Trewa/** del entregable de PTw@ndA al directorio donde se encuentra el servidor WildFly.

2. Por cada dataSource de Trew@ correspondiente a una instalación del motor de tramitación será necesario crear un directorio dentro de la carpeta solr, renombrado con el nombre indicado en la propiedad **pool-name** del datasource. Por ejemplo para un dataSource con propiedad **pool-name="TrewaDS_1"** deberá existir un directorio contenido en solr con el nombre **trewads_1 (siempre en minúscula).**

   Utilizar los directorios de ejemplo proporcionados trewads_1, trewads_2 renombrándolos en función de los nombres de los datasources.

   No es necesario crear un core de Solr para el dataSource por defecto TrewaDS.

3. Editar el fichero core.properties ubicado dentro de la carpeta creada en el paso anterior, editando las dos propiedades definidas:

   name=trewads_1

   collection=trewads_1

2.3.8.3.2 Configuración Solr en modo clúster (SolrCloud)

Solr 8.0.0 incluye una nueva función de alta disponibilidad llamado SolrCloud, que utiliza fragmentación de índices y replicación del esquema para lograr la persistencia de datos. Emplea Apache ZooKeeper para mantener la configuración y coordinar el estado entre los fragmentos.

El proyecto Solr mantiene una wiki señalando tres implementaciones básicas de SolrCloud. Para la aplicación simple, SolrCloud incluye una versión incorporada de ZooKeeper. Las configuraciones señaladas por el wiki aprovechan la aplicación embedded ZooKeeper. Aunque esto es aceptable para fines de desarrollo y pruebas, el rendimiento y alta disponibilidad nos llevan a la necesidad de una separación de Solr y ZooKeeper.

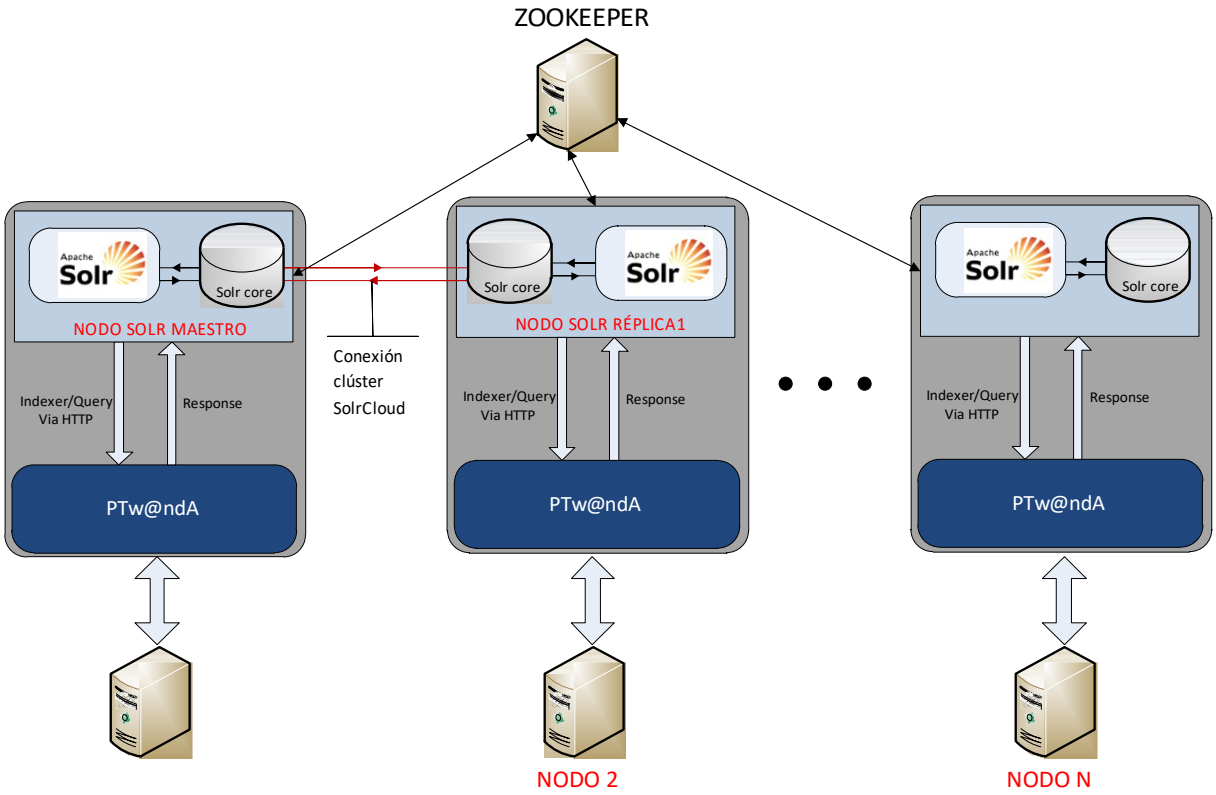
La implementación de SolrCloud propuesta para un entorno en modo clúster de PTw@ndA no hace uso de la funcionalidad de fragmentación de índices, utilizándose un factor de replicación 1, esto es, el conjunto de índices se almacena en un nodo maestro, existiendo otro nodo réplica para lograr la persistencia de datos.

Para una configuración en clúster de la plataforma formada por dos nodos, uno de los nodos de PT ejercería el rol de líder y el otro nodo el rol de réplica.

Infraestructura y arquitectura

La configuración recomendada para entornos de producción consta de lo siguiente:

- Un nodo Apache Zookeeper por cada nodo de PTw@ndA.
- Un único nodo Solr Maestro.
  - Un nodo réplica o más. En función del número de nodos que formen parte del clúster de PTw@ndA, se podrán configurar tantos nodos réplicas como nodos de PTw@ndA.



La configuración expuesta en la imagen anterior consta de un nodo zookeeper, encargado de coordinar los nodos de SolrCloud así como mantener los ficheros de configuración de Solr.

Para el clúster SolrCloud se han reutilizado dos nodos del clúster de PTw@ndA, desplegándose el componente Solr en el mismo servidor de aplicaciones, ejerciendo un servidor el rol de maestro y otro servidor el rol de réplica. En el caso de disponer de más de dos nodos para el clúster podrían configurarse varios nodos réplicas del nodo líder.

Zookeeper

A continuación se detalla los pasos a realizar para la configuración del servidor Zookeeper:

1. Copiar el directorio **zookeeper,** ubicado en la ruta **/05.Software base/Distribución_Zookeeper** del entregable de PTw@ndA, en un directorio de la máquina donde vaya a desplegarse zookeeper, por ejemplo, **/opt/zookeeper/.**
2. En el archivo de configuración zoo.cfg que se encuentra en /opt/zookeeper/conf/, indicar la ip y el puerto por defecto (2888:3888) de cada uno de los nodos de zookeeper que se van a configurar. Por ejemplo:

   server.1 = 7.128.80.48:2888:3888

   server.2 = 7.128.80.49:2888:3888

   server.3 = 7.128.80.50:2888:3888

3. En la carpeta **zookeeper_data** de cada uno de los nodos, hay que crear un archivo llamado myid que dentro tendrá el id que ocupa para zookeeper según la configuración anterior; por ejemplo, en el nodo 7.128.80.48 el archivo tendrá dentro el 1, en el nodo 7.128.80.49 el archivo tendrá el 2 y en el nodo 7.128.80.50 el archivo tendrá el 3.

4. Iniciar el nodo Zookeeper, ejecutando para ello el script siguiente:

   ./zookeeper/bin/zkServer.sh start

Es posible que en determinados versiones de algunos sistemas operativos sea necesario ejecutar el script de arranque con el usuario root. Si en el proceso de arranque se produce un error indicando que no se tienen permisos suficientes para crear ficheros en el directorio **zookeeper_data** será necesario ejecutarlo como **root.**

Se recomienda utilizar una configuración **en clúster para Zookeeper**, recomendándose un número mínimo de tres servidores zookeeper. http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_zkMulitServerSetup

SolrCloud

A continuación se detallan los pasos a seguir para configurar los nodos Solr, tanto el nodo maestro como el nodo réplica. Para ello se utilizarán dos servidores de aplicaciones WildFly 14.0.1 Final donde se encuentren desplegados ya dos nodos de la plataforma, reutilizándose el mismo servidor para ambas aplicaciones, PTw@ndA y Solr.

1. Copiar el directorio **solr-8.0.0** existente en la ruta **/05.Software base/Distribución_Solr/** del entregable de PTw@ndA al directorio **/opt**.

   En el caso de integrarse con más de una instancia del motor de tramitación Trew@ debe ejecutar las instrucciones del punto Soporte Multi-Trew@ para varios nodos. Luego volver al punto 2 de esta sección.

2. Incluir el parámetro de arranque de WildFly para indicar la ruta al directorio de solr copiado en el paso 1, desde la administración de WildFly o directamente en el fichero de configuración standalone. Para el caso de haber copiado el directorio dentro del /opt sería:

   solr.solr.home= /opt/solr-8.0.0/server/solr

3. Editar el fichero solr.xml del directorio /bin/solr de WildFly, configurando:
   a. La dirección ip y puerto del nodo zookeeper:

      <str name="zkHost">7.128.80.48:2181</str>

   b. Modificar puerto Solr estableciendo el puerto en el que se despliega dicho servidor:

      <int name="hostPort">8080</int>

   La configuración final del fichero solr.xml debe quedar de la siguiente manera:

```
<solrcloud>
       <str name="host">${jboss.bind.address}</str>
       <int name="hostPort">8080</int>
       <str name="hostContext">${hostContext:solr}</str>
       <int name="zkClientTimeout">35000</int>
       <bool name="genericCoreNodeNames">${genericCoreNodeNames:true}</bool>
       <str name="zkHost">7.128.80.48:2181</str>
 </solrcloud>
```

5. Ejecutar los script existentes en la ruta **/opt/solr-8.0.0/server/scripts/cloud-scripts/**, del entregable de PTw@ndA:

   zkcli.sh -cmd upconfig -zkhost **7.128.80.48:2181** -confdir **/ruta_WildFly/solr/collection1/conf/** -confname myconf

   zkcli.sh -cmd linkconfig -zkhost **7.128.80.48:2181** -collection collection1 -confname myconf -solrhome solr

   Los valores definidos en zkhost son las direcciones ip y puertos de los servidores zookeeper de cada uno de los nodos, mientras que el parámetro –confdir deberá contener la ruta al directorio conf, directorio que se encuentra dentro del directorio collection1 copiado en solr-8.0.0. Para el parámetro "myconf" ponemos el nombre del core por ejemplo.

   Para una instalación con más de una instancia del motor de tramitación Trew@ se deberá sustituir el nombre collection1 por el nombre del core de solr que hace referencia a cada jndi de las distintas instancias de Trew@.

   Es necesario editar la variable JAVA_HOME declarada dentro del script zkcli.sh.

6. Creación de los siguientes parámetros de configuración desde la Administración de PTw@nDA:
   a. **ACTIVO_SOLR_CLOUD:** Parámetro que indica si se ha realizado la configuración del componente SolrCloud. Para este caso, el valor a establecer será **true.**

b. **SOLR_ENDPOINTS:** URLs de cada uno de los nodos de solr, separados por una coma:

servidor1:8983,servidor2:8983

c. **ZOOKEEPER_ENDPOINTS:** URLs de los servidores zookeeper que contiene los ficheros de configuración de Solr, cuando la plataforma está desplegada en clúster:

servidor1:2181,servidor2:2181

7. Creación de nueva propiedad del sistemas

   Desde la consola de administración de WildFly se accede a la opción *System Properties* del menú *General Configuration* (pestaña Profile) y añadimos la siguiente propiedad:

   jboss.bind.addess = dirección_ip_nodo_servidor_WildFly

Finalizada la configuración se procederá al arranque de cada servidor, en el siguiente orden:

Para arrancar solr, será necesario ejecutar el siguiente comando:

./solr-8.0.0/bin/solr start cloud -z 7.128.80.48:2181, 7.128.80.49:2181, 7.128.80.50:2181 -noprompt

Si se estuviera como root, ha de añadirse el parámetro -force al comando.

Llegado a este punto ya se tendrá configurado SolrCloud. A continuación se crearán la/las collección/es necesaria/as. Para ello, habrá que:

1. Abrir la pestaña collections de la barra lateral izquierda.
2. Seleccionar la opción “Add Collection”.
3. Introducir el nombre de la colección y seleccionar la configuración establecida en el punto 4.
4. Indicar el número de fragmentaciones del índice a repartir entre los distintos nodos.(Se recomienda 1).
5. Indicar el el campo replicationFact el número de nodos existentes y pulsar “Add Collection”. Es decir, si estamos

trabajando con 2 nodos para Solr, en el campo replicationFact indicamos 2.

Se repetirá este paso para cada una de las colecciones que se deseen crear.

Una vez realizado los pasos anteriores, tendremos la aplicación configurada de la siguiente manera:

Cada core creado estará conformado por 1 shard (un fragmento de índices único y completo), el cual estará replicado en todos y cada uno de los nodos creados en el cluster de Solr.

En este caso, si tenemos un clúster configurado con 2 nodos, al iniciarse la aplicación, uno de los dos nodos ejercerá como líder de manera aleatoria, y si se cae, el otro nodo, al tener la misma información replicada, podrá ejercer como líder, y brindarnos de la información.

2.3.8.3.2.1 Soporte Multi-Trew@ para varios nodos

En el caso de desplegar la plataforma de tramitación para sincronizarse con más de una instancia del motor de tramitación Trew@ se deberán realizar los siguientes pasos:

1. Copiar el directorio **solr8.0.0** existente en la ruta**/05.Software base/Distribución_Solr/** del entregable de PTw@ndA al directorio **/opt/solr8.0.0**.
2. Por cada dataSource de Trew@ correspondiente a una instalación del motor de tramitación será necesario crear un directorio dentro de la carpeta solr, renombrado con el nombre indicado en la propiedad **pool-name** del datasource.

   Para ello coja la carpeta collection1 proporcionada en la ruta **05.Software base/Distribución_Solr/01. Solr_Mono_Trewa/solr-8.0.0/server/solr/**

   Por ejemplo para un dataSource con propiedad **pool-name="TrewaDS_1"** deberá existir un directorio contenido en solr con el nombre **trewads_1 (siempre en minúscula).**

   Utilizar los directorios de ejemplo proporcionados trewads_1, trewads_2 renombrándolos en función de los nombres de los datasources.

   No es necesario crear un core de Solr para el dataSource por defecto TrewaDS.

3. **Eliminar el fichero core.properties ubicado dentro de la carpeta creada en el paso anterior en caso de que se proporcione.**

2.3.8.4 Alta disponibilidad

2.3.8.4.1 Introducción

La configuración de un sistema de aplicaciones web en alta disponibilidad requiere siempre la redundancia en la instalación de los servidores de aplicaciones. Existen dos modos de hacer uso de esta redundancia:

- Se dispone de un nodo activo y uno o más nodos en espera pasiva. Habrá de habilitarse un sistema que permita detectar problemas de funcionamiento en el nodo activo y haga entrar en tal circunstancia en funcionamiento alguno de los nodos pasivos.
- Se dispone de dos o más nodos activos que se reparten (mediante un mecanismo hardware o software) la carga del trabajo. Con esta arquitectura, un sistema externo a estos nodos detecta los posibles problemas que puedan presentar; en tal caso, el nodo afectado dejará de recibir trabajo hasta su recuperación.

De las dos posibilidades planteadas es preferible, con diferencia, la segunda, ya que además de lograr la alta disponibilidad y, por tanto, la continuidad del servicio, se logra un mayor aprovechamiento de los recursos (no quedan recursos ociosos como en el primer caso) y se producen mejoras de rendimiento y su progresiva escalabilidad con la incorporación de más nodos activos al sistema. Por todo ello, en este documento se explicará cómo conseguir una infraestructura de este tipo para el despliegue del sistema PTw@ndA.

2.3.8.4.2 Problemas a resolver

- **Acceso local al sistema de archivos:**

  El despliegue de diversos servidores de aplicaciones en máquinas distintas puede suponer problemas si estas aplicaciones escriben datos en el sistema de archivos local de los equipos. Es obvio que esta escritura de datos no estará coordinada y originará problemas.

  En el caso que nos ocupa de PTw@ndA, no existe ningún problema con esta casuística.

- **Persistencia de sesiones de usuario:**

  Las aplicaciones web almacenan la información de sesión de la memoria volátil de los servidores. Si no se presta atención a este problema se darán situaciones de pérdida de sesión de los usuarios finales, ya sea por mala gestión del reparto de la carga del trabajo o por la caída no controlada de nodos del sistema.

2.3.8.4.3 Solución persistencia de sesiones de usuario

Para este problema existen dos soluciones que se detallan a continuación:

- El balanceador de reparto de carga de trabajo entre nodos debe identificar la sesión a la que pertenece cada una de las peticiones que ha de gestionar. Todas las peticiones de una misma sesión serán redirigidas siempre a un único nodo.

  Del mismo modo, se deberá detectar la creación de nuevas sesiones para asignar nodo de ejecución a los nuevos clientes. Generalmente, las aplicaciones web basadas en Java identifican la

sesión mediante el envío al navegador cliente de una cookie de sesión única con el nombre JSESSIONID vinculada a la ruta y al servidor de la aplicación.

Es necesario destacar en este punto que la caída de un nodo del sistema provocaría la reasignación a otro nodo de los usuarios conectados, implicando la pérdida de sesión sobre los mismos.

- Existe la alternativa de que todos los nodos intercambien mensajes de los datos de sesión que manejan. De este modo, no es necesario gestionar ninguna asignación de nodo a los clientes ni éstos se ven afectados por la caída de alguno de ellos. Como contrapartida, esta solución implica una mayor carga de trabajo y tráfico de red para los servidores de aplicaciones.

2.3.8.4.4 Solución ejemplo basada en Apache

Se plantea a continuación una solución al problema expuesto basada en el servidor web APACHE como elemento software balanceador de carga entre nodos. En nuestro caso, los distintos servidores de aplicaciones WildFly con PTw@ndA instalado.

Aunque no se detallará en este documento, es también posible la configuración de varios servidores web APACHE y realizar sobre ellos un balanceo con un sistema externo de alta disponibilidad (hardware o software).

La vinculación entre APACHE y WildFly se realizará con el módulo `mod-jk` de APACHE que permite vincularlo a cualquier servidor de aplicaciones que implemente el protocolo AJPv13 (igualmente, de la Fundación Apache).

2.3.8.4.5 Definición de worker.properties

En primer lugar será necesario identificar todos los nodos donde ha sido desplegado WildFly mediante un fichero de texto plano que denominaremos `worker.properties`. En él se definirán nuestros “workers”, donde un “worker” es un servidor o conjunto de servidores. Se supone que los servidores WildFly escuchan en el puerto por defecto APJ13, el puerto 8009.

```
# Listado de workers públicos definidos en este fichero
worker.list=ptw


# Se definirá un worker de tipo host para cada instalación de WildFly
# Definimos un primer nodo WildFly
worker.ptw1.host=nodo1.midominio.es
worker.ptw1.port=8009
worker.ptw1.type=ajp13
worker.ptw1.lbfactor=1
worker.ptw1.socket_timeout=35
worker.ptw1.recycle_timeout=10


# Definimos un segundo nodo WildFly
worker.ptw2.host=nodo2.midominio.es
```

```
worker.ptw2.port=8009
worker.ptw2.type=ajp13
worker. ptw2.lbfactor=1
worker.ptw2.socket_timeout=35
worker.ptw2.recycle_timeout=10

# Se podrían definir más nodos y asignarles pesos de reparto de carga
# haciendo uso de worker.ptwN.lbfactor

# Definimos el worker global ptw de tipo lb load balancing
worker.ptw.type=lb
worker.ptw.balance_workers=ptw1,ptw2
worker.ptw.sticky_session=false
worker.ptw.session_cookie=JSESSIONID
```

2.3.8.4.6 Inclusión de worker.properties en la configuración de APACHE

En la configuración de APACHE será necesario incluir y cargar el módulo mod_jk. También se establecerá el vínculo con el fichero "worker.properties" que habremos ubicado previamente en el directorio ${APACHE_HOME}/conf.

```
Include conf/mod-jk.conf
LoadModule jk_module modules/mod_jk.so
JkWorkersFile conf/workers.properties
JkLogFile logs/mod_jk.log
JkLogLevel info
```

Finalmente, se establecerán los puntos de montaje del contexto de PTw@ndA. Esta inclusión se realizará en el lugar apropiado del fichero de configuración, podrá estar en un Host Virtual o a nivel global en función de la particularidad de cada instalación.

```
JkMount /PTw@ndA* ptw
```

2.3.8.4.7 Configuración de los nodos WildFly

Todos los servidores de aplicación WildFly con PTw@ndA instalado deberán estar agrupados en una partición WildFly. Estos nodos deberán encontrarse ubicados en una misma VLAN que permita el intercambio de mensajes UDP entre ellos.

Es posible la construcción de una partición WildFly con servidores instalados en diferentes VLAN pero, en este caso, el intercambio de mensajes UDP no es posible, por lo que ha de efectuarse mediante TCP/IP, lo que requiere una configuración avanzada que no será cubierta en este documento.

Para la configuración en clúster de servidores WildFly 15.0.1 es necesario arrancar el servidor de aplicaciones utilizando la configuración apropiada. Para ello es necesario editar el fichero standalone.conf, modificando el valor de la propiedad **-Djboss.server.default.config=standalone-ha.xml.**

Una vez reiniciado el servidor de aplicaciones será necesario realizar la configuración del servidor para el modo standalone-ha seleccionado, para ello realizar los pasos descritos en el punto **3.3.3.2.1** y **3.3.3.2.3**.

Será necesario añadir dos nuevas propiedades del sistema, configurando el identificador del nodo que se está configurando y el nombre identificativo del clúster. Para ello se añadirá la siguiente propiedad:

jboss.node.name= *nombre identificativo del nodo* (Este valor debe ser único para cada nodo).

jboss.mod_cluster.jvmRoute = 1 (Este valor debe ser único para cada nodo)

jboss.partition.name= *nombre_identificativo_cluster (Este valor debe ser compartido por todos los nodos del clúster)*

El identificador establecido para el nodo en la propiedad anterior es necesario configurarlo en el fichero standalone-ha.xml, ubicado en la ruta **WildFly-15.0.1-Final\standalone\configuration.** Para ello se edita el subsistema siguiente añadiendo la propiedad I**nstance-id:**

```
<subsystem xmlns="urn:jboss:domain:undertow:3.1" instance-id="wildfly_ptwanda_253">
        <buffer-cache name="default"/>
        <server name="default-server">
            <ajp-listener name="ajp" socket-binding="ajp"/>
            <http-listener name="default" socket-binding="http" redirect-socket="https" enable-http2="true"/>
                <https-listener name="https" socket-binding="https" security-realm="ApplicationRealm" enable-
http2="true"/>
            <host name="default-host" alias="localhost">
                <location name="/" handler="welcome-content"/>
                <filter-ref name="server-header"/>
                <filter-ref name="x-powered-by-header"/>
            </host>
        </server>
        <servlet-container name="default">
            <jsp-config/>
            <websockets/>
        </servlet-container>
        <handlers>
            <file name="welcome-content" path="${jboss.home.dir}/welcome-content"/>
        </handlers>
        <filters>
            <response-header name="server-header" header-name="Server" header-value="WildFly/10"/>
                    <response-header name="x-powered-by-header" header-name="X-Powered-By" header-
value="Undertow/1"/>
        </filters>
    </subsystem>
```

Para evitar conflictos de conexión al clúster de otros servidores de aplicaciones WildFly 10 que se encuentren desplegados en la misma LAN debe modificarse la dirección multicast por defecto configurado en el fichero standalone-ha.xml. Para ello se edita la siguiente propiedad:

```
<socket-binding-group name="standard-sockets" default-interface="public" port-offset="${jboss.socket.binding.port-offset:0}">
    <socket-binding name="management-native" interface="management" port="${jboss.management.native.port:9999}"/>
    <socket-binding name="management-http" interface="management" port="${jboss.management.http.port:9990}"/>
    <socket-binding name="management-https" interface="management" port="${jboss.management.https.port:9443}"/>
    <socket-binding name="ajp" port="8009"/>
    <socket-binding name="http" port="8080"/>
    <socket-binding name="https" port="8443"/>
    <socket-binding name="jgroups-diagnostics" port="0" multicast-address="224.0.75.75" multicast-port="7500"/>
    <socket-binding name="jgroups-mping" port="0" multicast-address="${jboss.default.multicast.address:230.0.0.14}" multicast-port="45700"/>
    <socket-binding name="jgroups-tcp" port="7600"/>
    <socket-binding name="jgroups-tcp-fd" port="57600"/>
    <socket-binding name="jgroups-udp" port="55200" multicast-address="${jboss.default.multicast.address:230.0.0.14}" multicast-port="45688"/>
    <socket-binding name="jgroups-udp-fd" port="54200"/>
    <socket-binding name="modcluster" port="0" multicast-address="224.0.1.105" multicast-port="23364"/>
    <socket-binding name="osgi-http" interface="management" port="8090"/>
    <socket-binding name="remoting" port="4447"/>
    <socket-binding name="txn-recovery-environment" port="4712"/>
    <socket-binding name="txn-status-manager" port="4713"/>
    <outbound-socket-binding name="mail-smtp">
        <remote-destination host="localhost" port="25"/>
    </outbound-socket-binding>
</socket-binding-group>
```

El valor establecido para la dirección multicast deberá ser el mismo para todos los nodos que formen parte del clúster.

2.3.8.5 Configuración de trazas mediante Log4j

Para configurar la generación de mensajes de información, avisos y error del sistema mediante el sistema Log4j de Apache, debe editar el archivo **`log4j.properties`** que se encuentra dentro de la carpeta ***`[APLICACIÓN]\WEB-INF\CLASSES\`***.

Se considera que la configuración distribuida por defecto es válida para la mayoría de los entornos y no son requeridas tareas de configuración. En caso de que la configuración por defecto no satisfaga las necesidades de una instalación particular, este fichero puede ser adaptado, facilitándose a continuación un enlace a un manual de configuración del fichero **`log4j.properties`**.

2.4 Instalación de módulos funcionales. Cambio de versión.

Se ha previsto un mecanismo para simplificar las actualizaciones a futuras versiones de PTw@ndA. Así, para las nuevas versiones no será necesario volver a desplegar todos los componentes verticales instalados en versiones o implantaciones previas, sino que éstos se descargarán automáticamente en la nueva implantación desde el esquema de BBDD de PTw@ndA preexistentes.

Para que esta funcionalidad se lleve a cabo es necesario tener configurados 2 propiedades de la aplicación (más información en el apartado *Configuración de la aplicación*):

DESCARGAR_MODULOS = true.

APLICACION_SINCRONISMO_TRAMITADOR = true.

Una vez configurado esto, la primera vez que arranque la nueva implantación, una vez desplegado el WAR y configurados los DataSource, se accederá a BBDD, comprobándose los módulos instalados previamente e instalándose aquellos que no se encuentren.

También se ha implementado un control de versiones de tal forma que a la hora de desplegar PTw@ndA, si los módulos que se encuentran en BBDD no coinciden en nombre y tamaño con los desplegados se sustituirán por los existentes en BBDD.

Cada vez que se actualice o instale un módulo, será necesario reiniciar el servidor de aplicaciones para que los cambios tengan efecto y no quede inestable el contexto de la aplicación.

Esta funcionalidad es también útil para el despliegue de PT en situaciones de granjas de servidores en la que los nuevos nodos a incorporar desplegarán de modo automatizado todos los módulos verticales disponibles para el resto de los nodos que comparten el mismo esquema de base de datos de PT.

2.5 Definición de menús.

En esta nueva versión de PTw@ndA, se ha incluido una facilidad de mantenimiento de menús, mediante la cual definir un menú/submenú que permitirá gestionar los accesos mostrados desde la pantalla principal.

Tras una nueva instalación de la versión de PTw@ndA, no se mostrará ningún menú en la pantalla principal, por lo que será necesaria su definición tras el correcto despliegue de la aplicación.

Para más información referente al mantenimiento de Menús, consultar el punto "*4.1 Mantenimiento de Menús"* y "*4.2 Mantenimiento de Instalaciones"* del *Manual de Administración de PTw@ndA*.